Begin "synthesize a scfg"

Initialization

/* *See Figure 8A* */

en = SCFG entry node 1003

op = outermost process 1000
process_1000.state = Runnable
process_1000.runningThread = none
process_1000.runningPredecessors += (en, -)

tt = outermost thread 1001
process_1000.threads += thread_1001

fn = fork(1)_1002 /* *the first node of topological sort* */

tt.stateVariable = fn.index /* *Set default value of thread_1001's state variable to the value 1* */

thread_1001.cnodes += fork(1)_1002 /* *put fork(1) in thread_1001* */

fork(1)_1002.pthreads = thread_1001 /* *make fork(1)'s thread be thread_1001* */
fork(1)_1002.state = Suspended /* *make state of fork(1) be Suspended* */

1. Main Loop: First Iteration

cn = fork(1)_1002 /* *result of first "for" loop assignment to cn* */

sn = copy node cn and its expression to SCFG 1004

th = cn.pthreads = fork(1).pthreads = thread_1001

1.b Execution of code block B

switch to thread thread_1001

1.b.switchTT Execution of switch to thread

thread_1001.process = process_1000
process_1000.pthreads = empty /* *op does not belong to a thread, since it's outermost* */

p = thread_1001.process = process_1000

p.state == Runnable /* *due to initialization of process_1000* */

rn = new restart node 1005 which tests the thread_1001.stateVariable /* *Abbreviated as "th_1001.stateVar" in Figure 8A* */

run cnode process_1000 as snode rn_1005

1.b.switchTT.runCAS Execution of run cnode process_1000 as snode rn_1005

/* *snp loops over each SCFG node pointed to by process_1000.runningPredecessors* */

/* *process_1000.runningPredecessors contains en_1003 from Initialization* */

/* *therefore, an edge 1006 is created from en_1003 to rn_1005. Since predecessor edge 1065 has no label, edge 1006 is given no label.* */

/* *Since op.restartPredecessor is empty from Initialization, run cnode p as snode rn ends at this point* */

1.b.switchTT Execution of switch to thread continues

/* *See Figure 8B* */

/* *thread_1001.cnodes contains fork(1)_1002 from Initialization* */

cn = fork(1)_1002 /* *per only iteration of "for" loop in this invocation of switch to thread* */

fork(1)_1002.restartPredecessor = rn_1005 /* *establish a restartPredecessor edge 1007 from fork(1)_1002 to rn_1005, with edge 1007 being labeled with the topological index (of value 1) of fork(1)_1002* */

fork(1)_1002.state = Runnable

process_1000.state = Running
process_1000.runningThread = thread_1001

1.b Execution of code block B

run cnode cn as snode sn /* *cn has been set to fork(1)_1002 by first iteration of main loop; sn has been set to 1004 by first code block after main "for" loop* */

1.b.runCAS Execution of run cnode cn as snode sn

/* *Since fork(1) 1002 has no runningPredecessors this "for" loop is not iterated over* */

/* *Since fork(1) 1002 has a restartPredecessor indicated by edge 1007 as being rn 1005, an edge 1066 is created from rn 1005 to sn 1004, with the edge 1066 being labeled by the label on edge 1007.* */

1.b Execution of code block B

th.cnodes -= cn; /* *cn has been set to fork(1) 1002, so this statement removes 1002 from thread tt since it is no longer needed.* */

1.c Execution of code block C

/* *See Figure 8C* */

process = new process /* *This new process is process_1008. This is the process that represents fork(1), and this process will be around as long as there are threads of fork(1) yet to run.* */

process.state = Runnable /* *process_1008 is given a state of Runnable* */

process.runningThread = none /* *process_1008 is given no runningThread* */

process.runningPredecessors += (sn, -) /* *Establish an edge 1013 from process_1008 to sn 1004 with label = none.* */

th.cnodes += process /* *process_1008 is put inside thread_1001* */

/* *"for" loop iterates over each successor of cn (where cn has been set to fork(1) by the main "for" loop)* */

cns = conditional_A(2)_1016 /* *conditional_A(2)_1016 with topological ordering 2, is the first successor iterated over* */

thread = new thread /* *create a new thread, thread_1009, within which the thread that begins with conditional_A(2) will operate.* */

process.threads += thread */* Add thread_1009 to process_1008 */*

thread_1009.stateVariable = conditional_A(2)_1016.index */* The state variable of thread_1009 is initialized to the default value of 2. */*

thread.cnodes += cns; */* Put conditional_A(2) inside thread_1009 as a cnode of the thread which could be executed next */*

put cnode cns in thread thread/* *conditional_A(2) has its thread indicated as 1009 */*

cns.state = Suspended

cns = conditional_B(4) */* conditional "B," with topological ordering (4), is the second successor to fork(1) iterated over */*

thread = new thread */* create a new thread, thread_1010, within which the thread that begins with conditional_B(4) will operate. */*

process.threads += thread */* Add thread_1010 to process_1008 */*

thread.stateVariable = conditional_B(4)_1017.index */* The state variable of thread_1010 is initialized to the default value of 4. */*

thread.cnodes += cns; */* Put conditional_B(4) inside thread_1010 as a cnode of the thread which could be executed next */*

put cnode cns in thread thread/* *conditional_B(4) has its thread indicated as 1010 */*

cns.state = Suspended

2. Main Loop: Second Iteration

cn = conditional_A(2)_1016 */* result of second "for" loop assignment to cn */*

sn = copy node cn and its expression to SCFG */* see conditional_A 1011 of Figure 8C */*

th = cn.pthreads = the thread of conditional_A(2) */* Thread of conditional_A(2) set to 1009 by code block C above */*

2.b Execution of code block B

switch to thread thread_1009

2.b.switchTT Execution of switch to thread on thread 1009

/ th.process = process 1008; process_1008.pthreads = thread_1001 */*

2.b.switchTT.switchTT Recursive execution of switch to thread on thread 1001

/ Basically, this recursive execution just makes sure that thread_1001, which contains the thread_1009 that is about to be set to a Running state, is itself already in a Running state. */*

/ thread_1001.process = process_1000; process_1000.pthreads = empty */*

p = thread_1001.process = process_1000 = op;

/ process_1000.state= Running and process_1000.runningThread = thread_1001, so no need to call "suspend any running thread in process p" */*

/ since process_1000.state is not Runnable, no further action is taken by this recursive execution of switch to thread */*

2.b.switchTT Execution of switch to thread on thread 1009

/ Now return to setting thread_1009 as running since it has been confirmed that its containing thread_1001 is already running */*

p = thread_1009.process = process_1008;

p.state == Runnable */* due to previous execution of code block C in first main loop iteration, process_1008 is not already Running so there is no other running thread within it which would need to be suspended */*

rn = new restart node 1012 which tests thread_1009.stateVariable

run cnode process_1008 as snode rn _1012

2.b.switchTT.runCAS Execution of run cnode process 1008 as snode rn 1012

/ snp loops over each SCFG node pointed to by the runningPredecessors of cn (which is process_1008) */*

/ process_1008.runningPredecessors contains fork_1004 */*

/ create the edge 1014 from 1004 to rn_1012. Edge 1014 is given no label, just as predecessor edge 1013 has no label. */*

/ Since process_1008.restartPredecessor is empty from code block C, run cnode process_1008 as snode rn_1012 ends at this point */*

2.b.switchTT Execution of switch to thread continues

/ See Figure 8D */*

/ th.cnodes, which is thread_1009.cnodes, contains conditional_A(2)_1016 */*

cn = conditional_A(2)_1016 */* per only iteration of "for" loop in this invocation of switch to thread */*

cn.restartPredecessor = (rn_1012, conditional_A(2)_1016.index) */* Establish a restartPredecessor edge 1015 from conditional_A(2)_1016 to rn_1012, with edge 1015 being labeled with the topological index value 2 of conditional_A(2)_1016. */*

cn.state = Runnable */* Set the state of conditional_A(2) 1016 to Runnable */*

p.state = Running */* Set state of process_1008 to Running */*
p.runningThread = th */* Set runningThread of process_1008 to be thread_1009 */*

2.b Execution of code block B

run cnode conditional_A(2)_1016 as snode sn_1011

2.b.runCAS Execution of "run cnode conditional A(2) 1016 as snode sn 1011"

/ Since conditional_A(2) 1016 has no runningPredecessors this "for" loop is not iterated over */*

/ Since conditional_A(2) 1016 has a restartPredecessor indicated by edge 1015 as being rn 1012, an edge 1018 is created from rn 1012 to sn 1011, with the edge labeled by the label on edge 1015 */*

2.b Execution of code block B

th.cnodes -= cn; */* cn has been set to conditional_A(2) 1016, so this statement removes 1016 from thread_1009 since it is no longer needed. */*

2.d Execution of code block D

/ See Figure 8E */*

/ "for" loop over successors of conditional_A(2) */*

/ cns is first set to successor join(8) */*

th.cnodes += cns */* thread_1009 has node join(8)_1019 */*

put cnode cns in thread th */* join(8) 1019 is put on thread_1009 */*

cns.runningPredecessors += (sn_1011, use edge condition from conditional_A(2) to join(8) of accfg) */* Edge 1020 is given "CA1" as its edge label value */*

/ cns is secondly set to successor emit_B(3) */*

th.cnodes += cns */* thread_1009 has node emit_B(3) 1021 */*

put cnode cns in thread th */* emit_B(3) 1021 is put on thread_1009 */*

cns.runningPredecessors += (sn_1011, use edge condition from conditional_A(2) to emit_B(3) of accfg) */* Edge 1022 is given "CA2" as its edge label value */*

3. Main Loop: Third Iteration

cn = emit_B(3) */* result of third "for" loop assignment to cn */*

sn = copy node cn and its expression to SCFG <See 1023>

th = cn.pthreads = thread of emit_B(3) 1021 */* Thread of emit_B(3) 1021 set to 1009 by code block D above */*

3.b Execution of code block B

switch to thread thread_1009

3.b.switchTT Execution of switch to thread on thread 1009

/ th.process = process 1008; process_1008.pthreads = thread_1001 */*

3.b.switchTT.switchTT Recursive execution of switch to thread on thread 1001

/ Basically, this recursive execution just makes sure that thread_1001, which contains the thread_1009, is itself already in a Running state. */*

/ thread_1001.process = process_1000; process_1000.pthreads = empty */*

p = thread_1001.process = process_1000 = op;

/ process_1000.state= Running and process_1000.runningThread = thread_1001, so no need to call "suspend any running thread in process p" */*

/ since process_1000.state is not Runnable, no further action is taken by this recursive execution of switch to thread */*

3.b.switchTT Execution of switch to thread on thread 1009

/ Now return to confirming thread_1009 as running since it has been confirmed that its containing thread_1001 is already running */*

p = thread_1009.process = process_1008;

p.state == Running */* due to previous execution of code block B in second main loop iteration, process_1008 is already Running */*

p.runningThread == thread_1009 */* due to previous execution of code block B in second main loop iteration, process_1008 already has thread th as its runningThread */*

/ rest of switch to thread does nothing since p.state does not equal Runnable */*

3.b Execution of code block B

run cnode cn as snode sn */* cn has been set to emit_B(3) 1021 by third iteration of main loop; sn has been set to sn 1023 by first code block after main "for" loop */*

3.b.runCAS Execution of "run cnode emit_B(3) 1021 as snode sn 1023"

/ Since emit_B(3) 1021 has 1011 in its runningPredecessors, as indicated by edge 1022, an edge 1024 is added from 1011 to 1023. Edge 1024 is given the same label as predecessor edge 1022 (which has edge label value "CA2"). */*

/ Since emit_B(3) 1021 has no restartPredecessor, the rest of run cnode as snode is skipped */*

3.b Execution of code block B

th.cnodes -= cn; */* cn has been set to emit_B(3) 1021, so this statement removes 1021 from thread_1009 since it is no longer needed. */*

3.d Execution of code block D

/ See Figure 8F */*

/ "for" loop over successors of emit_B(3) */*

/ cns is first set to successor conditional_C(6) */*

th.cnodes += cns */* thread_1009 has node conditional_C(6)_1025 */*

put cnode cns in thread th */* conditional_C(6)_1025 is put on thread_1009 */*

cns.runningPredecessors += (sn, edge condition from emit_B(3) to conditional_C(6) in accfg) */* conditional_C(6) 1025 is given sn 1023 as its runningPredecessors, as indicated by edge 1026. Since the condition from emit_B(3) to conditional_C(6) in the accfg is none, no label is put on edge 1026. */*

/ There are no more successors to emit_B(3) */*

4. Main Loop: Fourth Iteration

cn = conditional_B(4) 1017 */* result of fourth "for" loop assignment to cn */*

sn = copy node cn and its expression to SCFG */* See conditional_B 1027 of Figure 8F */*

th = cn.pthreads = the thread of conditional_B(4) 1017 */* Thread of conditional_B(4) 1017 set to 1010 by code block C of Iteration 1 above */*

4.b Execution of code block B

switch to thread thread_1010

4.b.switchTT Execution of switch to thread on thread 1010

/ th.process = process 1008; process_1008.pthreads = thread_1001 */*

4.b.switchTT.switchTT Recursive execution of switch to thread on thread 1001

/ Basically, this recursive execution just makes sure that thread_1001, which contains the thread_1010, is itself already in a Running state. */*

/ thread_1001.process = process_1000; process_1000.pthreads = empty */*

p = thread_1001.process = process_1000 = op;

/ process_1000.state= Running and process_1000.runningThread = thread_1001, so no need to call "suspend any running thread in process p" */*

/ since process_1000.state is not Runnable, no further action is taken by this recursive execution of switch to thread */*

4.b.switchTT Execution of switch to thread on thread 1010

/ Now return to switching from thread_1009 to thread_1010 as running since it has been confirmed that the containing thread, for both 1009 and 1010, is already running */*

p = thread_1010.process = process_1008;

p.state == Running /* due to previous execution of code block B in second main loop iteration, process_1008 is already Running */

p.runningThread == thread_1009 /* due to previous execution of code block B in second main loop iteration */

/* Since process_1008.state == Running and process_1008.runningThread != thread_1010, suspend any running thread in process_1008 is executed */

4.b.switchTT.suspendART Execution of suspend any running thread in process_1008

p.state = Runnable; /* change state of process_1008 from Running to Runnable */

th = p.runningThread; /* set th to thread_1009, the previously running thread of process_1008 */

needToSaveState = true; /* Since thread_1009 has more than one node (namely, nodes 1019 and 1025) in thread_1009.cnodes. */

/* "for" iterates over each cnode in thread_1009.cnodes */

/* First iteration of "for" sets cn to join(8) 1019 */

/* join(8) 1019 is not a process */

/* join(8)_1019.runningPredecessors is not empty (since it points to 1011) */

sn = new save state node with assignment that "thread_1009.stateVariable = 8" /* Since needToSaveState is true, the assignment of 1028 is created. */

/* "for" loop sets snp to each runningPredecessor of join(8)_1019 */

/* only iteration of "for" loop sets snp to 1011 */

/* only iteration of "for" loop creates edge 1029 from 1011 to 1028. Edge 1029 is given a label value taken from the predecessor edge of 1020. */

/ See Figure 8G */*

/ process_1008 has 1028 added to its runningPredecessors list by edge 1030. No label value is given to edge 1030. */*

/ fork(8)_1019 has no restartPredecessor */*

fork(8)_1019.state = Suspended;

/ Second iteration of "for" sets cn to conditional_C(6)_1025 */*

/ conditional_C(6)_1025 is not a process */*

/ conditional_C(6)_1025.runningPredecessors is not empty (since it points to 1023) */*

sn = new save state node with assignment that "thread_1009.stateVariable = 6" */* Since needToSaveState is true, the assignment of 1031 is created. */*

/ "for" loop sets snp to each runningPredecessor of conditional_C(6)_1025 */*

/ only iteration of "for" loop sets snp to 1023 */*

/ only iteration of "for" loop creates edge 1032 from 1023 to 1031, where edge 1032 has no label since edge 1026 has no label. */*

/ See Figure 8H */*

/ process_1008 has 1031 added to its runningPredecessors list by arch 1033. */*

/ conditional_C(6)_1025 has no restartPredecessor */*

conditional_C(6)_1025.state = Suspended;

process_1008.runningThread = none; */* thread_1009 has been stopped, but thread_1010 has not yet been started */*

4.b.switchTT Re-execution of switch to thread on thread 1010

/ Now process_1008 is Runnable rather than Running */*

rn = new restart node testing thread_1010.stateVariable /* *Create restart node 1034* */

run cnode process_1008 as snode rn_1034

4.b.switchTT.runCAS Run cnode process 1008 as snode rn 1034

/* *"for" loop iterates over each runningPredecessor of process_1008* */

snp = 1028 /* *first runningPredecessor* */

/* *create edge 1035 from 1028 to rn_1034 which has no label, just as 1030 has no label* */

snp = 1031 /* *second runningPredecessor* */

/* *create edge 1036 from 1031 to rn_1034 which has no label, just as 1033 has no label* */

4.b.switchTT Re-execution of switch to thread on thread 1010

/* *See Figure 8I* */

cn = conditional_B(4)_1017 /* *first and only iteration of "for" loop* */

cn.restartPredecessor = (rn_1034, conditional_B(4)_1017.index) /* *indicated by edge 1037 from conditional_B(4)_1017 to rn_1034* */

cn.state = Runnable

process_1008.state = Running

process_1008.runningThread = thread_1010

4.b Execution of code block B

run cnode cn as snode sn /* *cn has been set to conditional_B(4) 1017 by fourth iteration of main loop; sn has been set to sn 1027 by first code block after main "for" loop* */

4.b.runCAS Execution of "run cnode conditional_B(4) 1017 as snode sn 1027"

/ Since conditional_B(4) 1017 has no runningPredecessors, this loop is skipped */*

/ Since conditional_B(4) 1017 has a restartPredecessor, an edge 1038 is added from rn_1034 to 1027. The condition of edge 1038 is taken from the label of edge 1037. */*

4.b Execution of code block B

th.cnodes -= cn; */* cn has been set to conditional_B(4) 1017, so this statement removes 1017 from thread_1010 since it is no longer needed. */*

4.d Execution of code block D

/ See Figure 8J */*

/ "for" loop over successors of conditional_B(4) 1017 */*

/ cns is first set to successor join(8) */*

th.cnodes += cns */* thread_1010 has node join(8)_1039 */*

put cnode cns in thread th */* join(8)_1039 is put on thread_1010 */*

cns.runningPredecessors += (sn, edge condition from condition_B(4) to join(8) in the accfg) */* join(8)_1039 is given 1027 as its runningPredecessors, as indicated by edge 1040. Edge 1040 is given edge label value "CB1". */*

/ cns is secondly set to successor emit_C(5) */*

th.cnodes += cns */* thread_1010 has node emit_C(5)_1041 */*

put cnode cns in thread th */* emit_C(5)_1041 is put on thread_1010 */*

cns.runningPredecessors += (sn, edge condition from condition_B(4) to emit_C(5) in the accfg) */* emit_C(5)_1041 is given 1027 as its runningPredecessors, as indicated by edge 1042. Edge 1042 is given edge label value "CB2". */*

5. Main Loop: Fifth Iteration

cn = emit_C(5) */* result of fifth "for" loop assignment to cn */*

sn = copy node cn and its expression to SCFG */* See emit_C 1043 */*

th = cn.pthreads = the thread of emit_C(5) 1041 */* Thread of emit_C(5) 1041 set to 1010 by code block D above */*

5.b Execution of code block B

switch to thread thread_1010

5.b.switchTT Execution of switch to thread on thread 1010

/ th.process = process 1008; process_1008.pthreads = thread_1001 */*

5.b.switchTT.switchTT Recursive execution of switch to thread on thread 1001

/ Basically, this recursive execution just makes sure that thread_1001, which contains the thread_1010, is itself already in a Running state. */*

/ thread_1001.process = process_1000; process_1000.pthreads = empty */*

p = thread_1001.process = process_1000 = op;

/ process_1000.state= Running and process_1000.runningThread = thread_1001, so no need to call "suspend any running thread in process p" */*

/ since process_1000.state is not Runnable, no further action is taken by this recursive execution of switch to thread */*

5.b.switchTT Execution of switch to thread on thread 1010

/ Now return to confirming thread_1010 as running since it has been confirmed that its containing thread_1001 is already running */*

p = thread_1010.process = process_1008;

p.state == Running */* due to previous execution of code block B in second main loop iteration, process_1008 is already Running */*

p.runningThread == thread_1010 /* due to previous execution of code block B in fourth main loop iteration, process_1008 already has thread th as its runningThread */

/* rest of switch to thread does nothing since p.state does not equal Runnable */

5.b Execution of code block B

run cnode cn as snode sn /* cn has been set to emit_C(5)_1041 by fifth iteration of main loop; sn has been set to sn 1043 by first code block after main "for" loop */

5.b.runCAS Execution of "run cnode emit_C(5) 1041 as snode sn 1043"

/* Since emit_C(5) 1041 has 1027 in its runningPredecessors, as indicated by edge 1042, an edge 1044 is added from 1027 to 1043. Label of edge 1044 is taken from edge 1042. */

/* Since emit_C(5) 1041 has no restartPredecessor, the rest of run cnode as snode is skipped */

5.b Execution of code block B

th.cnodes -= cn; /* cn has been set to emit_C(5) 1041, so this statement removes 1041 from thread_1010 since it is no longer needed. */

5.d Execution of code block D

/* See Figure 8K */

/* "for" loop over successors of emit_C(5) */

/* cns is set to only successor join(8)_1039 */

th.cnodes += cns /* thread_1010 already has node join(8)_1039 */

put cnode cns in thread th /* join(8)_1039 is already on thread_1010 */

cns.runningPredecessors += (sn, edge condition from emit_C(5) to join(8) in the accfg) /* join(8)_1039 has 1043 added to its runningPredecessors, as indicated by edge 1067. Edge 1067 has no label since there is no edge condition from emit_C(5) to join(8) in the accfg. */

/ There are no more successors to emit_C(5) */*

6. Main Loop: Sixth Iteration

cn = conditional_C(6) 1025 */* result of fourth "for" loop assignment to cn */*

sn = copy node cn and its expression to SCFG <See 1045>

th = cn.pthreads = the thread of conditional_C(6) 1025 */* Thread of conditional_C(6) 1025 set to 1009 by code block D of Iteration 3 above */*

6.b Execution of code block B

switch to thread th */* switch to thread 1009. Note we are changing from thread 1010 */*

6.b.switchTT Execution of switch to thread on thread 1009

/ th.process = process 1008; process_1008.pthreads = thread_1001 */*

6.b.switchTT.switchTT Recursive execution of switch to thread on thread 1009

/ Basically, this recursive execution just makes sure that thread_1001, which contains the thread_1009, is itself already in a Running state. */*

/ thread_1001.process = process_1000; process_1000.pthreads = empty */*

p = thread_1001.process = process_1000 = op;

/ process_1000.state= Running and process_1000.runningThread = thread_1001, so no need to call "suspend any running thread in process p" */*

/ since process_1000.state is not Runnable, no further action is taken by this recursive execution of switch to thread */*

6.b.switchTT Execution of switch to thread on thread 1009

/ Now return to switching from thread_1010 to thread_1009 as running since it has been confirmed that the containing thread, for both 1009 and 1010, is already running */*

p = thread_1009.process = process_1008;

p.state == Running */* due to previous execution of code block B in fourth main loop iteration, process_1008 is already Running */*

p.runningThread == thread_1010 */* due to previous execution of code block B in fourth main loop iteration */*

/ Since process_1008.state == Running and process_1008.runningThread != thread_1009, suspend any running thread in process_1008 is executed */*

6.b.switchTT.suspendART Execution of suspend any running thread in process_1008

p.state = Runnable; */* change state of process_1008 from Running to Runnable */*

th = p.runningThread; */* set th to thread_1010, the previously running thread of process_1008 */*

restartNode = none;

needToSaveState = false; */* Since thread_1010 only one node (namely, node 1039) in thread_1010.cnodes. */*

/ "for" iterates over each cnode in thread_1010.cnodes */*

/ First iteration of "for" sets cn to join(8) 1039 */*

/ join(8) 1039 is not a process */*

/ join(8)_1039.runningPredecessors is not empty (since it points to 1027 and 1043) */*

/ Since needToSaveState is false, the "else" clause is executed. */*

snp = 1027 */* by "for" loop */*

/ process_1008 has 1027 added to its runningPredecessors list by edge 1046. Edge 1046 gets the same label value as edge 1040 (where edge 1040 was given label value "CB1"). */*

snp = 1043 */* by "for" loop */*

/ process_1008 has 1043 added to its runningPredecessors list by edge 1068. Edge 1068 gets no label value since edge 1067 had no label value. */*

/ fork(8)_1039 has no restartPredecessor */*

fork(8)_1039.state = Suspended;

process_1008.runningThread = none; */* thread_1010 has been stopped, but thread_1009 has not yet been started */*

restartNode == none */* nothing to do here */*

6.b.switchTT Re-execution of switch to thread on thread 1009

/ See Figure 8L */*

/ Now process_1008 is Runnable rather than Running */*

rn = new restart node testing thread_1009.stateVariable */* Create restart node 1047 */*

run cnode process_1008 as snode rn_1047

6.b.switchTT.runCAS Run cnode process 1008 as snode rn 1047

/ "for" loop iterates over each runningPredecessor of process_1008 */*

snp = 1027 */* first runningPredecessor */*

/ create edge 1048 from 1027 to rn_1047. Edge 1048 is given edge label value "CB1" from edge 1046. */*

snp = 1048 */* second runningPredecessor */*

/ create edge 1049 from 1043 to rn_1047. Edge 1049 is given no edge label value since edge 1068 had no label value. */*

6.b.switchTT Re-execution of switch to thread on thread 1009

/ See Figure 8M */*

cn = conditional_C(6)_1025 */* first iteration of "for" loop */*

cn.restartPredecessor = (rn_1047, conditional_C(6)_1025.index) */* indicated by edge 1050 from conditional_C(6)_1025 to rn_1047. Edge 1050 is given a label value of 6. */*

cn.state = Runnable

cn = join(8)_1019 */* Second iteration of "for" loop */*

cn.restartPredecessor = (rn_1047, join(8)_1019.index) */* indicated by edge 1051 from join(8)_1019 to rn_1047. Edge 1051 is given a label value of 8. */*

cn.state = Runnable

process_1008.state = Running

process_1008.runningThread = thread_1009

6.b Execution of code block B

run cnode cn as snode sn */* cn has been set to conditional_C(6) 1025 by sixth iteration of main loop; sn has been set to sn 1045 by first code block after main "for" loop */*

6.b.runCAS Execution of "run cnode conditional_C(6) 1025 as snode sn 1045"

/ Since conditional_C(6) 1025 has no runningPredecessors, this loop is skipped */*

/ Since conditional_C(6) 1025 has a restartPredecessor, an edge 1052 is added from rn_1047 to 1045. Edge 1052 is given the label value of edge 1050. */*

6.b Execution of code block B

th.cnodes -= cn; /* *cn has been set to conditional_C(6) 1025, so this statement removes 1025 from thread_1009 since it is no longer needed.* */

6.d Execution of code block D

/* *See Figure 8N* */

/* *"for" loop over successors of conditional_C(6) 1025* */

/* *cns is first set to successor join(8)* */

th.cnodes += cns /* *thread_1009 already has node join(8)_1019* */

put cnode cns in thread th /* *join(8)_1019 is already on thread_1009* */

cns.runningPredecessors += (sn, edge condition from conditional_C(6) to join(8) in accfg) /* *join(8)_1019 is given 1045 as its runningPredecessors, as indicated by edge 1052. Edge 1052 is given label value "CC1" since this is the edge condition from conditional_C(6) to join(8) in accfg.* */

/* *cns is secondly set to successor emit_D(7)* */

th.cnodes += cns /* *thread_1009 has node emit_D(7)_1053* */

put cnode cns in thread th /* *emit_D(7)_1053 is put on thread_1009* */

cns.runningPredecessors += (sn, edge condition from conditional_C(6) to emit_D(7) in accfg) /* *emit_D(7)_1053 is given 1045 as its runningPredecessors, as indicated by edge 1054. Edge 1054 is given label value "CC2" since this is the edge condition from conditional_C(6) to emit_D(7) in accfg.* */

7. Main Loop: Seventh Iteration

cn = emit_D(7) /* *result of seventh "for" loop assignment to cn* */

sn = copy node cn and its expression to SCFG /* *See emit_D 1055* */

th = cn.pthreads = the thread of emit_D(7) 1053 /* *Thread of emit_D(7) 1053 set to 1009 by code block D above* */

7.b Execution of code block B

switch to thread th */* switch to thread 1009 */*

7.b.switchTT Execution of switch to thread on thread 1009

/ th.process = process 1008; process_1008.pthreads = thread_1001 */*

7.b.switchTT.switchTT Recursive execution of switch to thread on thread 1001

/ Basically, this recursive execution just makes sure that thread_1001, which contains the thread_1009, is itself already in a Running state. */*

/ thread_1001.process = process_1000; process_1000.pthreads = empty */*

p = thread_1001.process = process_1000 = op;

/ process_1000.state= Running and process_1000.runningThread = thread_1001, so no need to call "suspend any running thread in process p" */*

/ since process_1000.state is not Runnable, no further action is taken by this recursive execution of switch to thread */*

7.b.switchTT Execution of switch to thread on thread 1009

/ Now return to confirming thread_1009 as running since it has been confirmed that its containing thread_1001 is already running */*

p = thread_1009.process = process_1008;

p.state == Running */* due to previous execution of code block B in sixth main loop iteration, process_1008 is already Running */*

p.runningThread == thread_1009 */* due to previous execution of code block B in sixth main loop iteration, process_1008 already has thread th as its runningThread */*

/ rest of switch to thread does nothing since p.state does not equal Runnable */*

7.b Execution of code block B

run cnode cn as snode sn /* *cn has been set to emit_D(7)_1053 by seventh iteration of main loop; sn has been set to sn 1055 by first code block after main "for" loop* */

7.b.runCAS Execution of "run cnode emit_D(7) 1053 as snode sn 1055"

/* *Since emit_D(7) 1053 has 1045 in its runningPredecessors, as indicated by edge 1054, an edge 1056 is added from 1045 to 1055. Edge 1056 is labeled with value "CC2" which is taken from runningPredecessor edge 1054.* */

/* *Since emit_D(7) 1053 has no restartPredecessor, the rest of run cnode as snode is skipped* */

7.b Execution of code block B

th.cnodes -= cn; /* *cn has been set to emit_D(7) 1053, so this statement removes 1053 from thread_1009 since it is no longer needed.* */

7.d Execution of code block D

/* *See Figure 8P* */

/* *"for" loop over successors of emit_D(7)* */

/* *cns is set to only successor join(8)_1019* */

th.cnodes += cns /* *thread_1009 already has node join(8)_1019* */

put cnode cns in thread th /* *join(8)_1019 is already on thread_1009* */

cns.runningPredecessors += (sn, edge condition from emit_D(7) to join(8) in accfg) /* *join(8)_1019 has 1055 added to its runningPredecessors, as indicated by edge 1057. There is no edge label.* */

/* *There are no more successors to emit_D(7)* */

8. Main Loop: Eighth Iteration

cn = join(8) /* *result of eighth "for" loop assignment to cn* */

sn = copy node cn and its expression to SCFG /* *See join 1058* */

th = cn.pthreads /* *Here there appears to be some ambiguity about which thread the join resides in. However, it does not matter whether thread 1009 or 1010 is selected since both threads reside in the same process, which is really the issue.* */

8.a Execution of code block A

p = th.process /* *p set to the process of thread_1009 which is process_1008* */

th = p.pthreads /* *th set to the thread of process_1008 which is thread_1001* */

switch to thread thread_1001

8.a.switchTT Execution of switch to thread thread_1001

thread_1001.process = process_1000

process_1000.pthreads = empty /* *No recursive call to switch to thread* */

p = th.process /* *p = thread_1001.process = process_1000* */

process_1000.state = Running
process_1000.runningThread == thread_1001

/* *process_1000 is Running, but its runningThread is thread_1001 so no need to suspend any running thread in process_1000* */

/* *Since process_1000 is Running, this invocation of "switch to thread thread_1001" does nothing further* */

8.a Execution of code block A

suspend any running thread in process_1008;

8.a.suspendART Execution of suspend any running thread in process_1008

process_1008 == Running;

/* *Must suspend the running thread_1009* */

process_1008 = Runnable;

th = process_1008.runningThread = thread_1009

restartNode = none;

needToSaveState = false; */* Since only cnode of thread_1009 is join(8)_1019 */*

cn = join(8)_1019 */* Only iteration of outermost "for" sets cn to thread_1009.cnodes */*

cn != process */* No need to suspend any running process in join(8)_1019 */*

join(8)_1019.runningPredecessors != empty */* in fact, it has 1045 and 1055 */*

/ Since needToSaveState == false, do the "else" clause */*

snp = rn_1045 */* First iteration of else's "for" finds rn_1045 as runningPredecessor of join(8)_1019 */*

process_1008.runningPredecessors += (rn_1045, take label from edge 1052) */* See edge 1060 which is given label value "CC1" from edge 1052 */*

snp = rn_1055 */* Second iteration of else's "for" finds rn_1055 as runningPredecessor of join(8)_1019 */*

process_1008.runningPredecessors += (rn_1055, take label from edge 1057) */* See 1061 which is given no label value since edge 1057 has no label value */*

/ join(8)_1019.restartPredecessor != none; has rn_1047 */*

restartNode = rn_1047;

join(8)_1019.state = Suspended;

process_1008.runningThread = none;

restartNode != none; */* Contains node rn_1047 */*

process_1008.runningPredecessors += (m_1047, -); */* Add edge 1059 to process_1008 and give it no label value */*

8.a Execution of code block A

/ Figure 8Q */*

run cnode process_1008 as snode 1058

8.a.runCAS Execution of run cnode process_1008 as snode 1058

snp = 1047 */* First iteration of "for" each process_1008.runningPredecessors */*

add edge 1062 from 1047 to 1058; */* Edge 1062 gets the label of edge 1059 */*

snp = 1045 */* Second iteration of "for" each process_1008.runningPredecessors */*

add edge 1063 from 1045 to 1058; */* Edge 1063 gets the label of edge 1060 */*

snp = 1055 */* Third iteration of "for" each process_1008.runningPredecessors */*

add edge 1064 from 1055 to 1058;

8.a Execution of code block A

thread_1001.cnodes -= process_1008

8.d Execution of code block D

/ Since join(8) has no successors, code block D does nothing */*

/ Since there are no further cnodes in topological sort, beyond join(8), main loop ends */*